Carnegie Mellon
Software Engineering Institute

Pittsburgh, PA 15213-3890

Requirements Management in a System-of-Systems Context: A Workshop

CMU/SEI-2006-TN-015

B. Craig Meyers
James D. Smith
Peter Capell
Patrick R. H. Place

March 2006

Integration of Software-Intensive Systems Initiative
Acquisition Support Program

This work is sponsored by the U.S. Department of Defense.

The Software Engineering Institute is a federally funded research and development center sponsored by the U.S. Department of Defense.

This work was created in the performance of Federal Government Contract Number FA8721-05-C-0003 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center. The Government of the United States has a royalty-free government-purpose license to use, duplicate, or disclose the work, in whole or in part and in any manner, and to have or permit others to do so, for government purposes pursuant to the copyright license under the clause at 252.227-7013.

For information about purchasing paper copies of SEI reports, please visit the publications portion of our Web site (http://www.sei.cmu.edu/publications/pubweb.html).

Table of Contents

List of Figures

Acknowledgements

We thank the attendees for their interest and participation in this work. We acknowledge insightful discussions with Cecilia Albert regarding this work. We also acknowledge Ira Monarch for participating in the workshop, as well as for many discussions. We also thank Patricia Oberndorf and Stephen Blanchette for comments.

Abstract

This report summarizes the results of a workshop focused on requirements management in a system of systems. The workshop attendees were affiliated with the Army Program Executive Office (PEO) Aviation and Training and Doctrine Command (TRADOC) Combat Developers. During the workshop, issues were identified in a number of areas, including requirements management, system-of-systems management, and system construction. Many of the issues raised address some form of the conflict that exists between a top-down, policy driven approach to the acquisition of a system of systems and a bottom-up, program-centric approach to the acquisition of an individual system.

1 Introduction

This note describes the results of a workshop focused on issues related to requirements management for a system of systems (SoS). The workshop was held at Ft. Rucker, Ga., on September 13–14, 2005. In the workshop, issues also surfaced about the acquisition and construction of an SoS. Some analysis of the issues was performed with the attendees during the workshop; however, most of the analysis was conducted following the workshop.

As background, a definition of the term *system of systems* is relevant; the following is from the *Joint Capabilities Integration and Development System* (JCIDS):

> **system of systems (SoS):** A set or arrangement of interdependent systems that are related or connected to provide a given capability. The loss of any part of the system will significantly degrade the performance or capabilities of the whole. The development of a[n] SoS solution will involve trade space between the systems as well as within an individual system performance. An example of a[n] SoS would be a combat aircraft. While the aircraft may be developed as a single system, it could incorporate subsystems developed for other aircraft. For example, the radar from an existing aircraft may be incorporated into the one being developed rather than developing a new radar. The SoS in this case would be the airframe, engines, radar, avionics, etc. that make up the entire combat aircraft capability [JCS 05].

A related term is *family of systems* (FoS); however, no distinction between these two terms was made at the workshop. These terms are further discussed in Appendix A.

This report is organized in the following manner:

- Section 2 describes the approach taken in the workshop.
- Section 3 presents the issues elicited and an initial categorization of them.
- Section 4 presents an analysis of the issues.
- Section 5 discusses the relation between the issues identified in this workshop and those of a previous workshop.
- Section 6 contains a brief summary of the report.
- Appendices
  - Appendix A examines the terms system of systems and family of systems.

- Appendix B provides an alternative view of requirements management in a system-of-systems context.
- Appendix C provides a brief list of typical activities associated with requirements management.
- Appendix D contains a discussion of workshop attendees regarding the Army Software Blocking policy.
- Appendix E provides an example in the development of patterns from the attendee issues.
- A list of acronyms is in Appendix F.

2 Workshop Approach

2.1 Purpose and Scope

The major purpose of the workshop was to elicit and analyze issues associated with requirements management for an SoS. A brief description of the traditional requirements management approach appears in Appendix C. Differences in approach are introduced when one considers an SoS; an alternative perspective on requirements in the system-of-systems context appears in Appendix B. Any distinction in scope of requirements discussed at the workshop was recognized by noting the context of use (e.g., a system-of-systems requirement or a system requirement).

The major activities of the workshop were to

- have the attendees develop a context diagram for requirements management—a diagram showing actors and their interactions[1]
- elicit attendee issues and then relate them to the context diagram

A simple example of a context diagram for the requirements management process for an SoS, as presented to the attendees, is shown in Figure 1. The example here is very simple in comparison to the actual system-of-systems environment; however, such simplification can provide useful insights, assuming the data used to construct the diagram is sufficiently grounded in real experience. This figure shows two program management organizations (PMOs), PMO-A and PMO-B, engaged in an acquisition. Each PMO has an associated milestone decision authority (MDA), and a prime contractor (possibly, subcontractors as well) for the development of the individual systems. The Joint Requirements Oversight Council (JROC) is also included because of its role in the requirements process. Thus, Figure 1 illustrates a context diagram that could be used to frame the discussion for issues identified by attendees.

[1] The term *actor* is used to indicate any organization or thing that affects requirements management; thus, actors may include people, organizations, standards, or commercial off-the-shelf (COTS) products. In other words, actors include more than stakeholders.

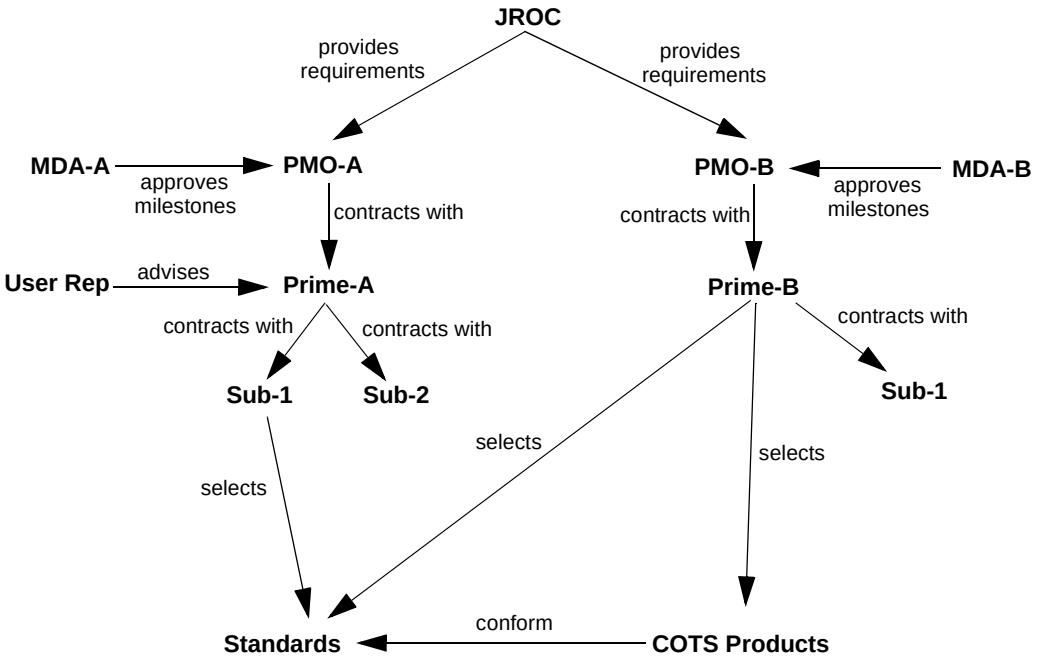


Figure 1: A Simple Context Diagram

Another purpose of the workshop was to identify interoperability considerations presented by the attendees. This was done in the context of the System-of-Systems Interoperability (SOSI) model. A key aspect of the SOSI model is that interoperability comes in various flavors. While traditionally considered in the context of an operational system, the SOSI model also includes programmatic aspects, as well as consideration of how system(s) are constructed.[2] This subject is discussed in *Proceedings of the System of Systems Interoperability Workshop (February 2003)* [Levine 03] and *System of Systems Interoperability (SOSI); Final Report* [Morris 04].

[2] Very loosely speaking, interactions between the PMOs are mainly an example of programmatic interoperability, while interactions between the prime contractors can be viewed as an example of constructive interoperability.

2.2 Attendees

The workshop attendees came from various U. S. Army (Army) organizations, reflecting significant breadth and depth of acquisition and user experience. Attendees were combat and materiel developers from USAAWC DCD and PEO Aviation.[3] These included active duty military, government civilian, and contractor personnel with a wide range of operational user and requirement system experience.

2.3 Initial Plan

The planned agenda for the workshop is summarized below:

- Introduction
- Attendee Presentations: allow attendees to present their organization contexts
- Background on Requirements Management: describe the role of requirements management
- Issue Elicitation: allow attendees to discuss their three most significant issues for prioritization by the group
- Context Discussion: identify actors, their relationships, and attendee development of context diagram(s)
- Relating Issues to Context Diagram: examine issues identified by attendees to see how issues relate to the context diagram
- Introduction to SOSI model: define a context for interoperability discussions
- Decomposition of Issues: use the SOSI model as a basis for discussing various issues related to interoperability regarding requirements management—with the goal of identifying the programmatic, constructive, and operational aspects
- Next Steps: enumerate the next steps recommended by the attendees
- Summary and Action Review

2.4 Conduct of the Workshop

The actual conduct of the workshop deviated from the plan for several reasons, including:

[3] A list of acronyms appears in Appendix F.

- During the attendee presentations, a number of issues were raised. However, the issues were not placed in an overall context that was explicit to all of the attendees.
- There were concerns about the context diagram. Several attendees pointed out that a full specification of the context diagram would be quite a large undertaking. For future workshops, an alternative might be to bring in a specification of the context diagram prior to conducting the workshop.
- A majority of the attendees had to leave after the first day, although several were kind enough to participate by a telcon on the second day.
- There was discussion of the Army software blocking policy at various times during the workshop. Such discussion was not included in the analysis, but the discussion points were captured and appear in Appendix D.

After completing about half of the planned agenda, we made a change. In particular, following issue elicitation and prioritization, we examined the higher priority issues. We sought to determine the actors relevant to an issue and the relations among the actors. While the term interoperability often came up, we held no discussion of the issues in terms of the various forms of interoperability. Despite the adjustment in approach, much valuable information was gathered. The issues raised are described in Section 3. There was insufficient time or attendees present to discuss possible next steps.

2.5 Considerations for Future Workshops

The need to change from the planned approach raises issues for the future conduct of workshops that are similar in intent to the one described here. In some sense, the planned approach had a top-down view, while the conduct of the workshop was bottom-up.[4] Among the issues regarding the conduct of the workshop, the following are relevant:

- *How should the context diagram be presented?* The context diagram helps to identify actors and their relations and allows for shared understanding of some domain. The original plan was to have such a diagram developed by the attendees to gain *their* perspective of the context. (There may be a difference between the ways things are supposed to be and the way they are!) As pointed out previously, there were concerns about the size and complexity of the con-

[4] We use the terms *top down* and *bottom up* to refer to alternative approaches. The former refers to the development of a context diagram for relevant aspects of acquisition prior to discussion of issues, and the latter refers to elicitation of issues prior to developing a context diagram.

text diagram. An alternative may be to have the attendees present their issues; the relevant context diagram can then be developed after the workshop.

- *Would an issue-first approach be viable?* Such an approach was followed at the workshop without the prior development of the context diagram. It seemed to work well in terms of the information elicited. The detriment of such an approach is the lack of an "integration harness" in which to address issues.
- *Should the workshop length be increased, perhaps to two or two-and-a-half days?* There is opportunity for information-gathering and analysis at the workshop. However, it is difficult for attendees to spend even more than a day at such an event.
- *Should the workshop be conducted as a series of interactions?* For example, the first interaction would be an elicitation and exploration of the issues. Then, the context diagram would be developed offline. A second interaction would be the presentation of the context diagram to the attendees, from which further discussion would ensue.

Regardless of the structure and conduct of the workshop, the intended goals must be kept in mind. Several choices are possible. One goal might be to develop the context diagram for some system-of-systems topic, such as requirements management. Another goal might be to identify issues relative to some particular topic. Still another goal might be some combination of those goals. In any case, an understanding of the issues is necessary to develop a solution approach.

3 Identification of Issues

The following list presents the issues as developed by the workshop attendees. The issues are presented in the form "<condition>; <consequence>." (Hence the character ";" may be read as "therefore.") The issues are listed in prioritized order, from highest to lowest, as determined by the attendees.

1. There is no system engineering staff above the program manager (PM) level (e.g., Army or DoD); this results in multiple solutions (stovepipes), suboptimized (at a lower level) for similar problems—systems-of-systems hierarchy is missing.
2. System-of-systems requirements are not clearly documented or configuration controlled/managed; they cannot be further allocated, derived, or met.
3. The system-of-systems user combat requirement developer is not clearly chartered; cannot derive a single set of requirements.
4. System-of-systems policy and mandates are without incentives and enforcement; does not ensure change from stovepipe to SoS.
5. The Planning, Programming, Budgeting and Execution System (PPBES) is not synchronized with the realities of system-of-systems acquisition; the resources required are not in the right place at the right time (event-driven versus a time-driven process conflict).
6. There is no method for validating and adjudicating interoperability requirements in the documentation process; interoperability requirements are not defined early or identified as a common development goal.
7. There is no *ownership* of system-of-systems requirements; there is no follow-up for their explanation or verification.
8. There is no single funding line for system of systems; cannot bring personnel, resources, and priorities together sufficiently to develop the common requirements.
9. The general system-of-systems procedure currently in place does not produce effective results; there is a need for a process that is comprehensive, to include time frame, management structure, definition of terms, results, and responsibilities.
10. The JCIDS process has no path that leads to a view (architecture) that can be used for a statement of specification for a material developer or test criteria

by the combat developer; there is no direct link from requirements to end product.

11. The Army has no tools (more automated than Excel, a protocol checker) to adequately model interoperability; must wait until done to achieve interoperability by trial and error.
12. There is a lack of application of system engineering to capability development and gap analysis (science and technology); multiple organizations (uncoordinated) working on multiple solutions to solve either the same problem or similar problems.
13. There are insufficient resources for requirements development and management; shortcuts are taken and key aspects may be missed.
14. System-of-systems lessons learned and "good idea" insertion are not incorporated and managed; the benefit of experience and impact of requirement creep are not considered.
15. DoD maintains multiple systems with independent users, requirements, and timelines with no single authority; there is no coordinated end product.
16. There is a Battle Command Migration Plan without a Network Migration Plan; impacts interoperability requirements process.
17. There is no comprehensive system-of-systems description available to all developers by which we can determine interfaces; we cannot ensure our designs achieve interoperability.
18. There is a lack of Joint Vision (e.g., Network-Centric Operations and Warfare Reference Model [NCOW-RM]) and a system-of-systems organization structure; leads toward a stovepipe development.
19. Some systems have *no* requirements documents (Tactical Operations Centers [TOCs] and Army Airborne Command and Control System [A2C2S]) or the documentation is out of date; there is no clear end state with which to coordinate.
20. The standards for knowledge, skills, and attributes of capability developers are unclear; the quality of documents may suffer.
21. The incentives for successful Command/PM do not support sacrificial resource distribution; system-of-systems interoperability will only be as good as its weakest proponent.
22. Joint system-of-systems requirements are not clear; interoperability is not guaranteed and joint testing results are questionable.
23. There is a lack of tradeoff analysis in the combat development process; software acquisition is not efficient, effective, and timely.

4 Analysis

The following subsections describe an analysis of the issues based on the system-of-systems requirements, system-of systems acquisition management, system-of-systems construction, and joint considerations. Most of the context diagrams (also known as *interoperability maps*) to be presented here were developed after the workshop. An attempt is also made to integrate the various context diagrams into one diagram.

4.1 Approach

The analysis of attendee issues was conducted with the goal of developing and understanding the context diagrams. (An example of a context diagram appeared in Figure 1.) The steps involved in the analysis included an identification of patterns and grouping of patterns into thematic constructs.

All of the elements of a context diagram, as shown in Figure 1, exhibit the same underlying form: They can be represented as a *pattern* whose structure is a tuple $<A_1, A_2; R>$, where A_1 and A_2 represent actors and R describes the relation between them. For example, as shown in Figure 2, the textual statement "The JROC provides requirements to the PMO" could be represented in a *diagrammatic form* or in terms of a (textual) pattern. (The relation is expressed as a verb phrase—here "provides requirements"; when parts of the phrase are obvious, a shorthand can be used.)

Figure 2: Representation of a Textual Statement

Patterns may express the negation of an actor, a relation among actors, or both. When a textual statement is described in terms of a negation, we use the term *antipattern*. In terms of a tuple, the general form for an antipattern will be denoted <$\underline{A_1}$, $\underline{A_2}$; $\underline{R}$> with the underscores denoting the negation of the actor(s) and/or their relation, as appropriate. For example, the statement "The PMO does not perform system engineering" would be represented as the antipattern: <PMO, system engineering; perform>.[5] (Notice, too that the statement "The PMO does not perform system engineering" actually has two interpretations. First, there is an understated assertion that the PMO *should* perform system engineering. The second interpretation is equivalent to the assertion that the PMO does *not* perform system engineering.)

The development of a context diagram is based on the use of patterns (and antipatterns). The patterns are developed from an examination of the textual material. An example of this process is shown in Appendix E on page 42.

Having collected individual patterns, the next step was to identify central themes on which the patterns could be aggregated. No formal process was applied to identify such themes. However, it became clear that the issues collected could be grouped into the following themes:

- system-of-systems requirements
- system-of-systems acquisition management
- system-of-systems construction
- joint issues

Each of these themes will be discussed in the following sections. The key results of the analyses will be integrated and presented in the summary of this section.

4.2 System-of-Systems Requirements

Since the primary goal of the workshop was to elicit and understand issues related to requirements management in a system-of-systems context, we will begin the analysis there. The issues deemed relevant to requirements include (read the ";" character as the word *therefore*):

- System-of-systems requirements are not clearly documented or configuration

[5] The antipattern <PMO, system engineering; perform> uses a shorthand notation for the relation: rather than as the more formal does perform, the relation is presented as perform.

controlled/managed; they cannot be further allocated, derived, or met (Issue 2).

- The system-of-systems user combat requirement developer is not clearly chartered; cannot derive a single set of requirements (Issue 3).
- There is no method for validating and adjudicating interoperability requirements in the documentation process; interoperability requirements are not defined early or identified as a common development goal (Issue 6).
- There is no *ownership* of system-of-systems requirements; there is no follow-up for their explanation or verification (Issue 7).
- There are insufficient resources for requirements development and management; shortcuts are taken and key aspects may be missed (Issue 13).
- Some systems have *no* requirements documents (TOCs and A2C2S) or the documentation is out of date; there is no clear end state with which to coordinate (Issue 19).
- The standards for knowledge, skills, and attributes of capability developers are unclear; the quality of documents may suffer (Issue 20).

Prior to addressing a context diagram for the issues, we developed with the attendees a small context diagram of the participants in the requirements process. This diagram is shown in Figure 3.

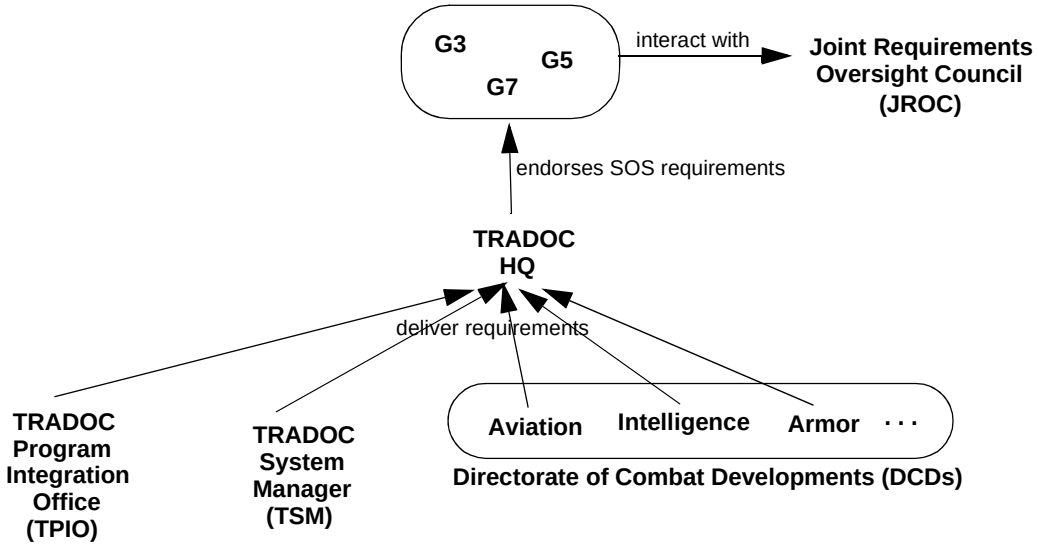


Figure 3: High-Level Context Diagram for Requirements

As illustrated in Figure 3, several organizations—such as a TPIO, TSM, or DCDs within TRADOC—are responsible for delivering requirements to TRADOC HQ.

In turn, TRADOC HQ must interact with various Army staff organizations (G3, G5, and G7) that endorse requirements for an SoS. The G3, G5 and G7 functions are combined: collectively, they form strategy, develop the force, manage the functional aspects of command and control, and establish requirements and priorities for the employment of the forces. Following this interaction, there is an interaction with the JROC.

The attendees emphasized that the process must also account for a bottom-up perspective because the details of the user requirements for individual platforms are only available at the "bottom." Overall, the process must provide for an integration of interoperability requirements from the top (e.g, JCIDS) with platform-focused requirements that come from a user program.[6]

A context diagram of the identified issues appears in Figure 4. The complexity of the diagram reflects the complexity of the issues identified. (Recall that an underlined term represents the negation of actors and/or their relations, as appropriate.)

Some points regarding this figure include

- A number of issues relate to staff: In particular, it was observed that there
  - were insufficient resources for requirements development
  - was a lack of standards for knowledge and skills of Capability Developers
  - was no charter for the user representative to participate in the process
- A number of issues relate to the activities associated with requirements management:
  - lack of methods for requirements validation and conflict adjudication
  - documentation of requirements
  - configuration control and management
- In addition, the following are worth noting:
  - Allocation of requirements to a particular system is difficult.
  - The quality of requirements suffers.

One inference that follows from the activities related to requirements management is that the requirements management process is not performed efficiently. This inference is supported by several of the issues identified by the attendees.[7]

---

[6] Notice that Figure 1 on page 4 indicates that JROC provides requirements to a PMO. Those requirements are presented in the form of an Initial Capabilities Document (ICD).

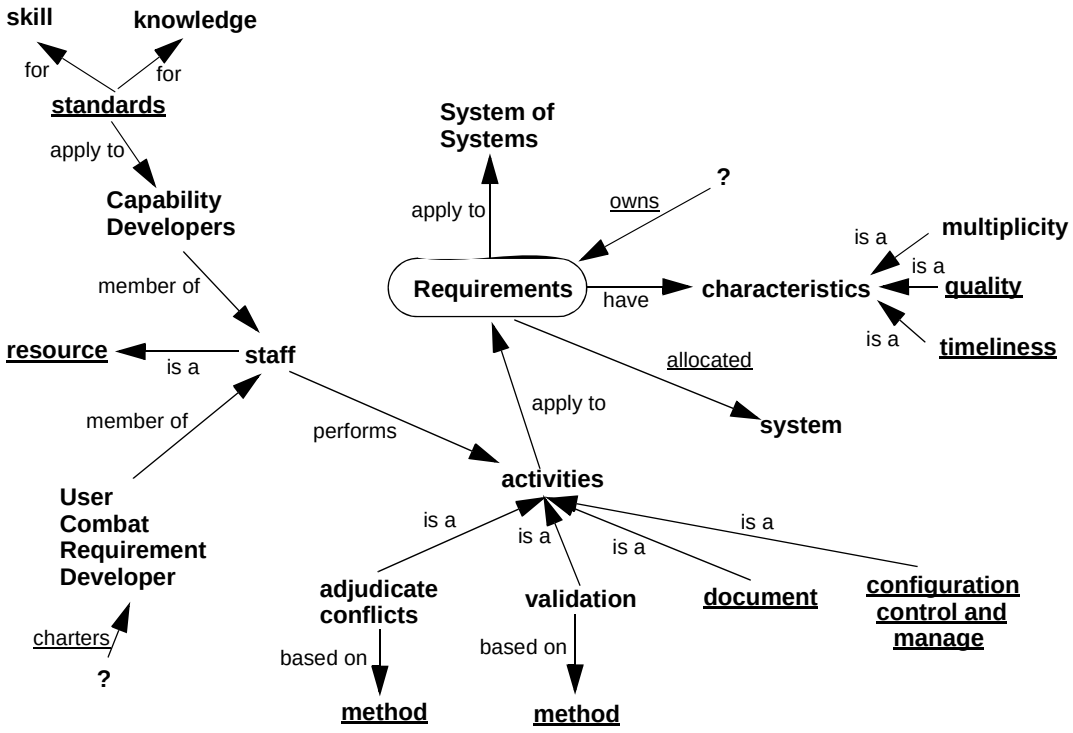


Figure 4: Requirements Context Diagram

4.3 System-of-Systems Acquisition Management

Another category of issues relates to the manner in which a system-of-systems acquisition is managed. The issues relevant to this category include:

- System-of-systems policy and mandates are without incentives and enforcement; does not ensure change from stovepipe to SoS (Issue 4).
- The PPBES is not synchronized with the realities of system-of-systems acquisition; the resources required are not in the right place at the right time (event-driven versus time-driven process conflict). (Issue 5)
- There is no single funding line for systems of systems; cannot bring personnel, resources, and priorities together sufficiently to develop the common requirements (Issue 8).
- The general systems-of-systems procedure currently in place does not produce

[7] The attendees did not identify the organization that would be responsible for providing the charter indicated in Figure 4—hence the "?". In addition it was not clear what organization "owned" the requirements for an SoS, although a number of possibilities were offered by the attendees.

effective results; there is a need for a process that is comprehensive, to include time frame, management structure, definition of terms, results, and responsibilities (Issue 9).

- Systems-of-systems lessons learned and "good idea" insertion are not incorporated and managed; the benefit of experience and impact of requirement creep are not considered (Issue 14).
- DoD maintains multiple systems with independent users, requirements, and timelines with no single authority; there is no coordinated end product (Issue 15).
- There is a Battle Command Migration Plan without a Network Migration Plan; impacts interoperability requirements process (Issue 16).
- The incentives for successful Command/PM do not support sacrificial resource distribution; system-of-systems interoperability will only be as good as its weakest proponent (Issue 21).
- There is a lack of tradeoff analysis in the combat development process; software acquisition is not efficient, effective, and timely (Issue 23).

A context diagram for the acquisition management of an SoS is shown in Figure 5.

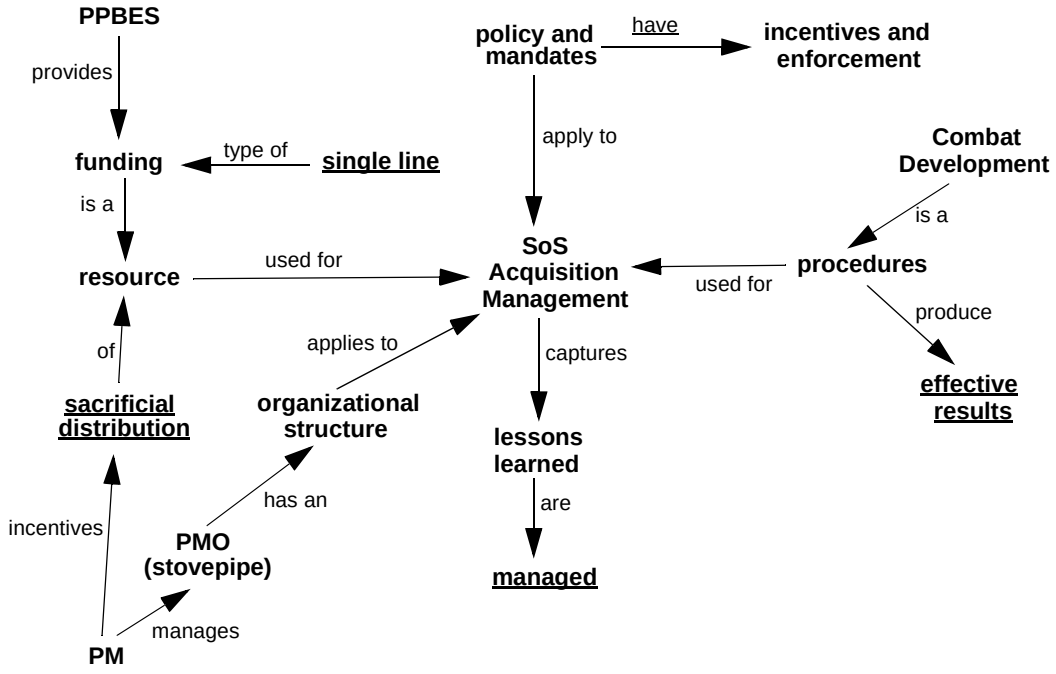


Figure 5: System-of-Systems Acquisition Context Diagram

The aspect of the management of an SoS with regard to funding is also apparent

in a number of these issues. A small, very simplified, context diagram was developed by the attendees, and it appears in Figure 6. The attendees did not identify the relation between the G3 and G8 organizations; hence the "?".

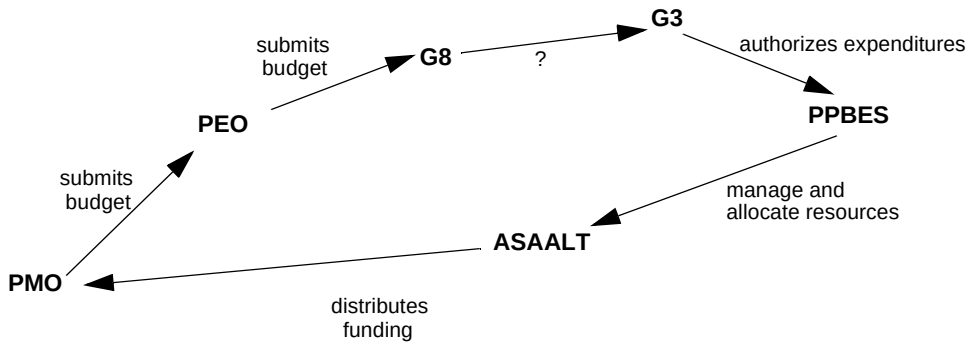


Figure 6: Context Diagram for Funding

A key actor in this discussion is the PPBES.[8] The purpose of the PPBES is to manage and allocate resources to the DoD. The process by which the PPBES operates is divided into phases, shown in Figure 7. The PPBES operates on a two-year cycle with multiple cycles running concurrently. The activities associated with PPBES are both generic to the DoD and specific to the Army, as shown in the figure.[9]

	Planning	Programming	Budgeting	Execution
DoD	Develop Defense Planning Guidance	OSD review of program objective memorandums and possible alternatives	Joint review of service budgets. Budget approval	Apply funds appropriated by Congress to approved programs
Army	Identify effective and efficient use of resources to establish capabilities	Translate planning guidance into a detailed allocation for a six-year period	Formulate, justify, and execute budget	Supervise and direct financial execution of approved budget

Figure 7: PPBES Phases

[8] Recall that an actor can be anything that participates in the requirements management process. Often, an actor is an organization or an individual, but this need not always be the case.

[9] Some background on the PPBES can be found at http://www.finance.army.mil/ppbes.htm.

An issue (Issue 5) identified during the workshop was that the PPBES was not synchronized with the realities of system-of-systems acquisition. In particular, the resources required for an SoS may not be in the right place at the right time. As noted by the attendees, another perspective on this question is to recognize the PPBES as a time-driven process, while major acquisition decisions are event driven. This is illustrated in Figure 8, which depicts the relation between the JCIDS process and the acquisition decisions (from *Operation of the Joint Capabilities Integration and Development System* [JCS 04]). The events at the top of Figure 8 are associated with the JROC process and those in the center are associated with program acquisition; the bottom of the figure is associated with the PPBES process. The crux of the issue relates to the synchronization of events of a variable nature (such as JCIDS and acquisition milestones) with events having a fixed timeline (those associated with the PPBES).

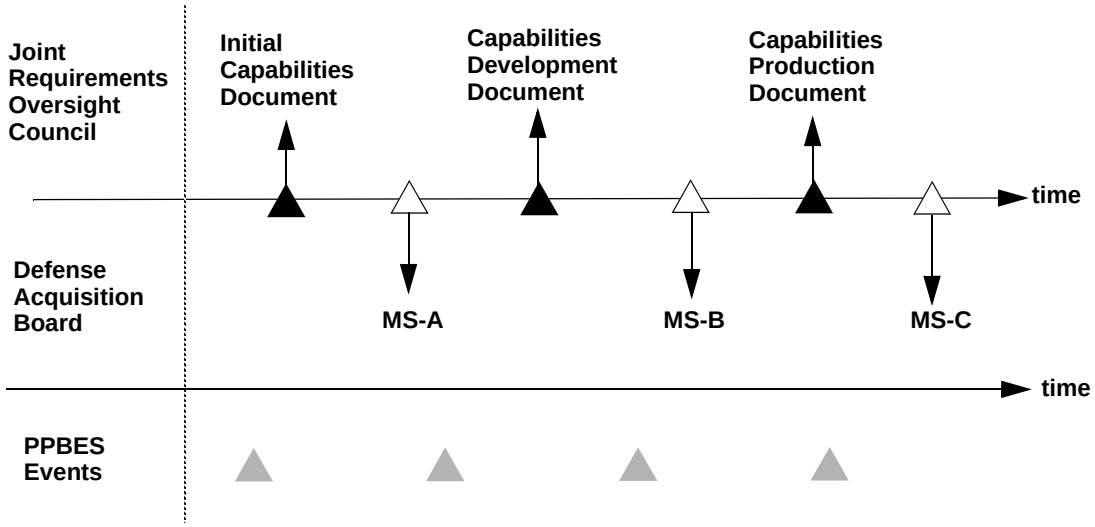


Figure 8: JCIDS, Acquisition, and PPBES Events

Moreover, and of relevance to the system-of-systems context, the superposition of *many* program events associated with the PPBES and their harmonization is at the heart of the issue. The myriad interactions between the acquisition system and the PPBES is another reflection of the bottom-up (i.e., program-centric) perspective on how the process is executed. On the other hand, a strength of the PPBES is that it updates information on all programs on an annual basis, regardless of where the acquisition programs are in their life cycle.

Issues associated with funding are related to other timelines. As pointed out by the attendees, a system may have a 30-year span, the development for that sys-

tem may take 10 years, and the typical tour of a program manager is 3 years. To this mix are added funding cycles of a 1- or 2-year duration. A depiction of the various life-cycle aspects is shown in Figure 9. In addition to the comments from the attendees, we have added the role of technologies and products that play a large part of the development of a system—or system of systems. This further illustrates the volatility that is a fundamental part of the acquisition process.

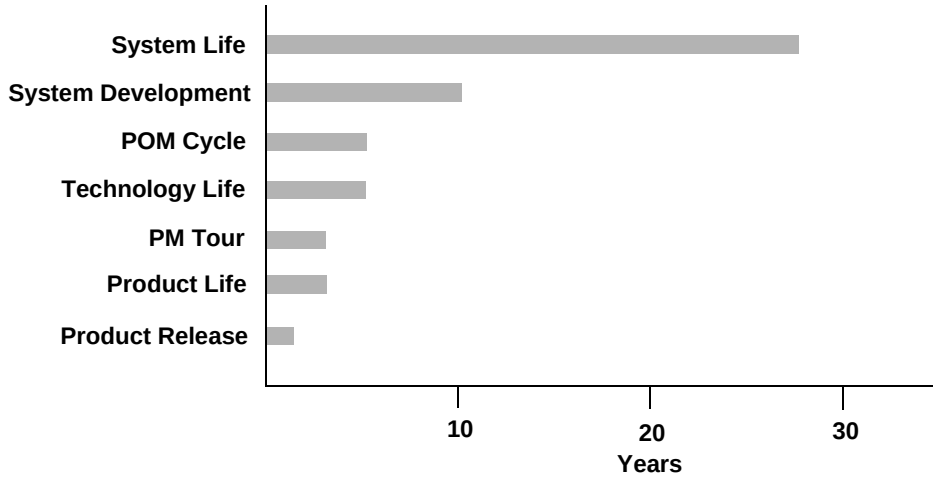


Figure 9: Relative Life-Cycle Durations

4.4 System-of-Systems Construction

The term *system construction* is used to refer to those activities that relate to the development of a system. Thus, *system-of-systems construction* refers to the construction of an SoS. The following issues relate to this topic.

- There is no system engineering staff above the PM level (e.g., Army or DoD); this results in multiple solutions (stovepipes), suboptimized (at a lower level) for similar problems—system-of-systems hierarchy is missing (Issue 1).
- The Army has no tools (more automated than Excel, a protocol checker) to adequately model interoperability; must wait until done to achieve interoperability by trial and error (Issue 11).
- There is a lack of application of system engineering to capability development and gap analysis (science and technology); multiple organizations (uncoordinated) working on multiple solutions to solve either the same problem or similar problems (Issue 12).
- There is no comprehensive system-of-systems description available to all developers by which we can determine interfaces; we cannot ensure our designs achieve interoperability (Issue 17).

The highest rated issue identified by the attendees was *There is no system engineering staff above the PM level (Army or DoD); this results in multiple solutions (stovepipes), suboptimized (at lower level) for similar problems—system-of-systems hierarchy is missing.* A context diagram showing the relation between actors engaged in the system engineering process is shown in Figure 10. This context diagram, like others presented previously, was based on an analysis of the issues identified. In this case, however, we have inferred the absence of an SoS System Engineering Organization, shown at the top of the figure. This inference was based on the identified lack of a system engineering staff above the PMO level. The lack of staff implies the lack of an associated organization.

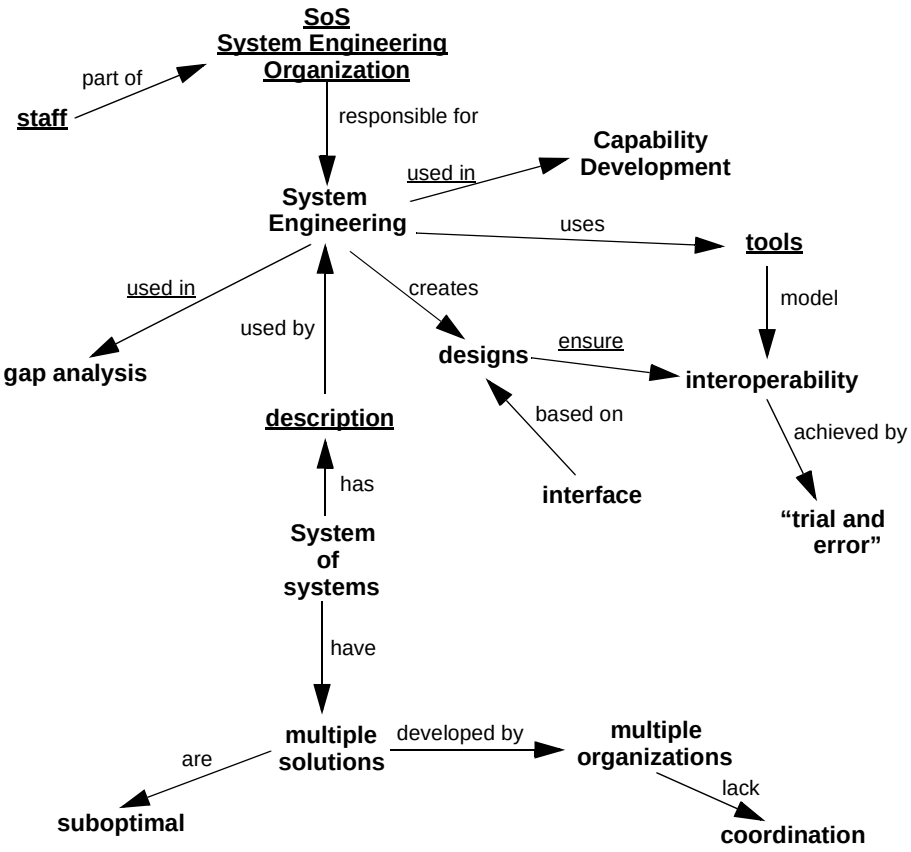


Figure 10: System-of-Systems Construction Context Diagram

Figure 10 also shows a thread related to the use of models to assess capabilities, part of an expected system engineering approach. In fact, the analysis of alternatives (AOA) was mentioned as not being done in the context of an SoS.

It is relevant to incorporate a discussion of the Army regulation that applies to this point. The following is from *Army Acquisition Policy*, regarding a PM's responsibilities:

> The systems engineering program will include full integration of the programs of other systems that have an interoperability requirement with this system. This includes both technical/operational synchronization and schedule harmonization across programs of interoperational systems. . . . Systems engineering must be done from a system-of-systems perspective in accordance with the Software Blocking Policy and its implemented processes [DoA 03, p. 40].

One infers from this statement that system engineering must be done in accordance with the software blocking policy [DoA 01]. Yet, the phrase "system engineering" does not appear in the software blocking policy. There are clear implications from this approach. In particular, each program manager is responsible for system engineering for their program, developed in the larger system-of-systems perspective. It is not clear, however, how conflicts are adjudicated in the process and who is responsible for the system-of-systems view as a whole.

4.5 Joint Considerations

There is a clear intent to move from a requirements-based process to one that is based on capabilities. This transformation is described in the *Joint Capabilities and Integration Development System (JCIDS) Process* and related documents [JCS 01, JCS 03, JCS 04, and JCS 05].

Although it was not the intent of this workshop to focus on issues related to joint requirements management, several issues were identified, including:

- The JCIDS process has no path that leads to a view (architecture) that can be used for a statement of specification for a material developer or test criteria by the combat developer; there is no direct link from requirements to end product (Issue 10).
- There is a lack of Joint Vision (e.g., NCOW-RM) and a system-of-systems organization structure; leads toward a stovepipe development (Issue 18).
- Joint system-of-systems requirements are not clear; interoperability is not guaranteed and joint testing results are questionable (Issue 22).

A context diagram for the issues related to the joint environment appears in Figure 11. Details of the approach used to construct this context diagram may be found in Appendix E.

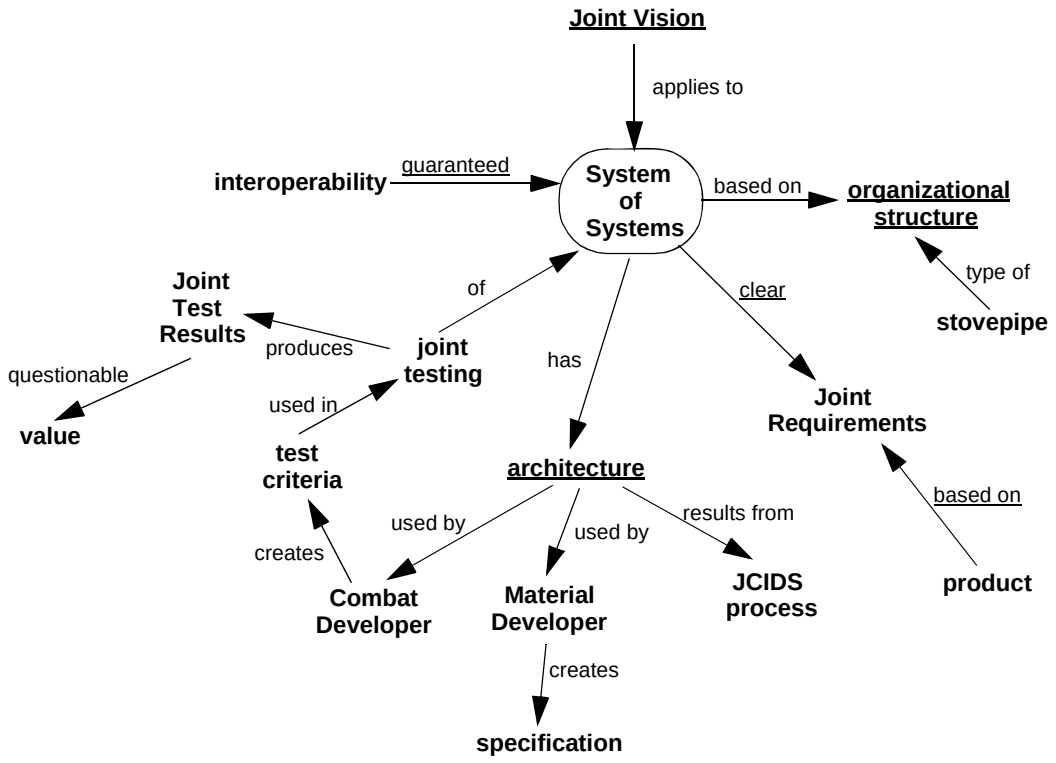


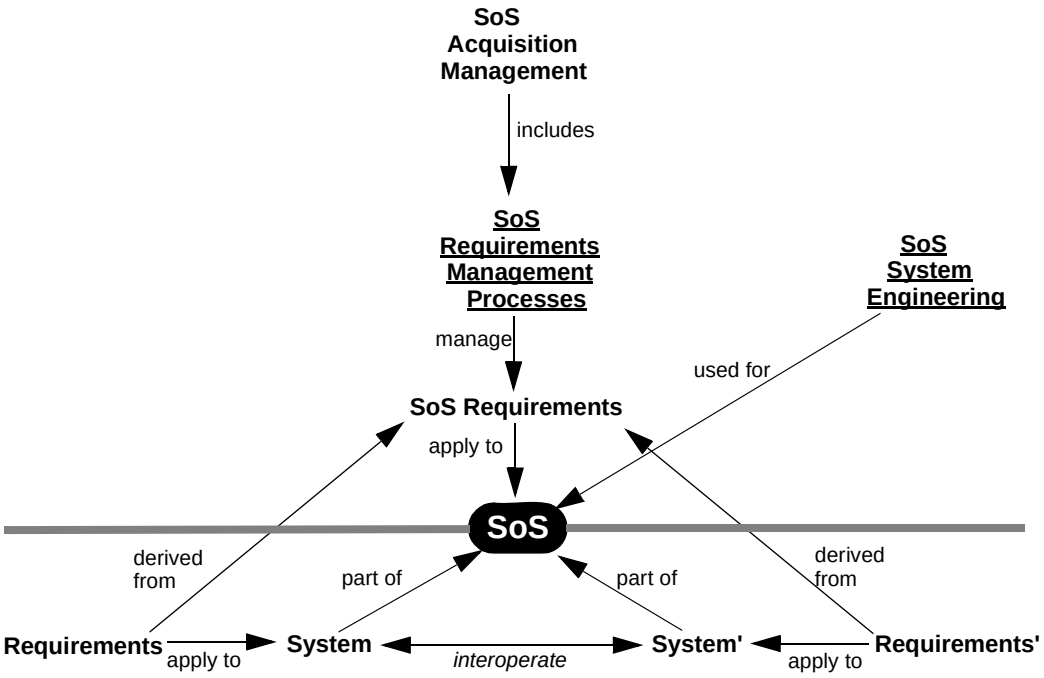
Figure 11: Context Diagram for Joint Issues

In some sense, the issues identified at the workshop for the joint environment are tantamount to a scaling of the Army-centric issues. For example, the identified lack of a system engineering process for an SoS within the Army will have implications for the success of system engineering in a joint system-of-systems environment. In addition, it was pointed out that there is a merger of top-down capabilities with a bottom-up perspective, based on requirements. Such a perspective can lead to difficulties, some of which appeared even in the context of Army Aviation issues identified in this workshop. Note also that the issue of a lack of clarity in specification of joint requirements will have an impact on the Army's ability to implement those requirements.

Because the focus of the workshop was on Army Aviation, rather than on joint considerations, we will not further address the joint issues. However, inferences from the preceding discussion may well apply to the joint scope. The integration of the JCIDS process with a service-specific process for some domain, such as requirements management, may be worth further consideration.

4.6 Summary of Analyses

Although the workshop was oriented toward a system-of-systems perspective, it is not possible to eliminate considerations of a PMO. In fact, such points were frequently raised in the discussion of the issues. Thus, we begin by considering a high-level context diagram shown in Figure 12.



Note: Shaded line denotes the boundary between a system-of-systems perspective and a PMO perspective.

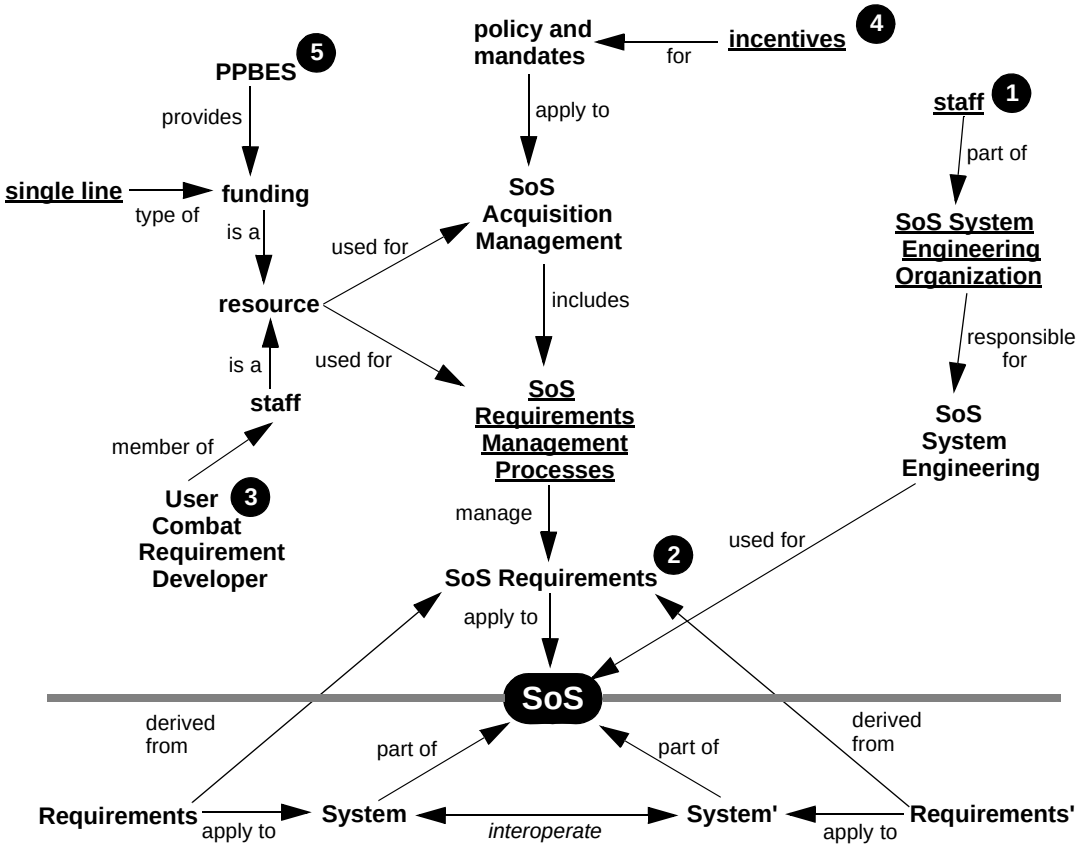
Figure 12: Themes for Integrated Issues

Consideration of the actors and their relations as shown in Figure 12 depicts two perspectives:

1. There is a program-centric view, shown at the bottom of the figure, where the programs apply a traditional system engineering approach to the development (and maintenance, etc.) of their systems. This part of the figure could have been elaborated more to show details of that approach.
2. There is a system-of-systems view, shown at the top of the figure, that can be interpreted as the application of a traditional system engineering approach to an SoS.

We now expand the view of Figure 12 to account for the issues identified by the attendees. Only the highest priority issues will be considered. Upon doing so we have the context map shown in Figure 13.

The view shown in Figure 13 reflects the concerns of the attendees and the issues they raised. (The issue numbers shown in circles refer to the list presented on page 8 and mentioned in this section.) Recall, for example, that several of the actors shown in this figure were inferred in order to frame the discussion. In fact, several such actors do not exist (e.g., the SoS Acquisition Management organization).



Note: Shaded line denotes the boundary between a system-of-system perspective and a PMO perspective. Numbers in circles denote issues.

Figure 13: Summary Context Map

Thus, while there is an attempt to apply a traditional system engineering view to the SoS (i.e., "after all, a system of systems is just another system"), there are conflicts in the way that this view is manifest.

Several points stand out, as noted by the attendees:

- There is no system engineering above the program level.
- Requirements are not managed in the context of the SoS.
- System-of-system policies and processes are not sufficiently comprehensive and are without mandates and incentives.
- There is no single funding line for an SoS.

On the one hand, there is the desire to treat an SoS as another system. But on the other hand, there are conflicts (funding, management, and system engineering, for example) that prevent such an approach from succeeding.

5 Relation to Future Force Workshop

The SEI conducted a workshop with individuals connected with the Future Force; the results of that session are presented in *Exploring Programmatic Interoperability: Army Future Force Workshop* [Smith 05]. The attendees of the Future Force Workshop came from various Army organizations and reflected a significant breadth and depth of acquisition experience, including individuals from Army headquarters; staff elements; Joint and Army acquisition organizations; operational test, research, and development areas; and training and doctrine groups.

In the following sections, we examine the issues identified by those individuals at that workshop. (We will also refer at times to the system-of-systems requirements management workshop as the *current* or *present* workshop.)

5.1 Organizational Concerns

The highest priority issue from the Future Force Workshop was the following: *The Army is not organized to develop a system of systems. There is a lack of understanding of requirements, money allocation, interaction, and testing.*

This issue results from the fact that while the Army fields operational capability as integrated units of personnel and equipment in a defined structural relationship, it procures systems individually, in response to separate operational requirements, appropriations, or the like. As a result, the organization of the acquisition system does not inherently encourage the tradeoffs, across systems within the SoS, necessary to maximize operational effectiveness of the Army as a whole.

Many of the results from the current workshop are directly related to the issue of organizational factors in the development of an SoS. For example, attendees identified issues related to a lack of

- system engineering staff above the PM level
- understanding of system-of-systems requirements
- a comprehensive process for an SoS to include time frame, management structures, and responsibilities

In the Future Force Workshop, participants also placed concerns related to

requirements under this topic. In the present workshop, these issues were explored in greater detail and confirmed the results obtained in the Future Force Workshop.

5.2 Procurement vs. Development Life Cycle

This was the second highest priority issue from the Future Force Workshop: *The procurement versus development life-cycle models cause interoperability problems when functions are implemented.*

This issue arises when different systems that have to be interoperable are procured separately. Interoperability is commonly defined by sets of standards and interfaces, with systems required to implement these in some common fashion. Frequently, the organization responsible for procuring system *A* chooses for programmatic reasons (e.g., funding profile) to implement pieces of the required standards in a different way than that selected by the acquiring organization for system *B* (for equally sound programmatic reasons).

The differences in implementation approach can result in the two systems being unable to interoperate until both have implemented all portions of the specified standards. Since the procurement life cycles for both systems are driven by their *individual* requirements and funding lines, interoperability is often delayed for unacceptably long periods of time.

Coupling the procurement and development life-cycle models raises issues pertaining to funding. In the current workshop, participants identified several concerns related to the lack of

- synchronization between the PPBES and the realities of the acquisition process
- a single funding line for an SoS
- resources for requirements development and management

Since procurement life cycles are associated with individual systems (having their own requirements and funding lines), synchronization of multiple system acquisitions only makes the problem more difficult. And this problem is further exacerbated by the lack of a system engineering function at the system-of-systems level.

5.3 Migration Plans

This was the third highest priority issue from the Future Force Workshop: *A*

migration plan must be at the appropriate higher level, not based on a bottom-up perspective. Network—not the radios, fielding.

This issue reflects the "disconnect" between the present approach to migration planning (i.e., system by system) and the need to plan migration at the force capability level (e.g., Future Force). Unresolved, this issue can lead to a decrease in interoperability if an upgraded system does not provide an exact "form-fit-function" replacement for its predecessor—as is often the case. One example cited by the participants involved a new system being fielded to operational units that were required to interoperate: as each unit received the new system—in accordance with the fielding plan for the new system—it had the old system removed. Unfortunately, the new system wasn't fully backward-compatible with the old system, and the system fielding plan didn't reflect the operational reality that the units had to deploy and work together. Until *all* the units received and installed the new system, then, there was a net loss of interoperability and, as a result, operational effectiveness.

In the present workshop, the topic of a migration plan came up but was of relatively lesser importance. In particular, it was pointed out there that a Battle Command Migration Plan without a Network Migration Plan can impact the interoperability requirements process. This view represents a different, though related, perspective from that raised in the Future Force Workshop.

Both groups' comments point to the need for a discussion of requirements for the migration plan. Further, such discussion must be at the higher level—not at a system or program level because that perspective leads to a locally optimized approach without consideration of the system-of-systems perspective.

5.4 Measuring Operational Benefit

The fourth highest priority issue from the Future Force Workshop was: *There is a need for a process for measuring operational benefit of proposed interoperability solutions (e.g., cost-benefit analysis).*

Because so much of the focus on justifying the upgrades and migration of new capability is driven by the individual system's cost-benefit analysis, there is no agreed-upon mechanism for performing such analyses at the system-of-systems level. Or, where such analyses are performed, they are frequently driven by the procurement, fielding, and sustainment costs of the proposed upgrade, not of the original system. This approach generally results in an AOA that reflects a locally optimized solution *for an individual program or system* rather than a measure of

the operational benefits of the proposed upgrades in the larger perspective.

Two items from this workshop are related to operational benefit.

1. There was a perceived lack of tradeoff analysis in the combat developer process, resulting in inefficient, ineffective, and untimely software acquisition. Note also the close relation between measuring operational benefit and migration planning. In particular, migration of system capabilities is driven by an individual system; however, this may not be the same perspective when one considers operational benefit at the system-of-systems level.
2. A second point made was that there was a lack of analysis of alternatives performed in the context of an SoS. This point was noted in connection with Figure 10 on page 19.

5.5 Summary

The most important overall result from the Future Force Workshop was a recognition that better understanding is needed of the relationship between the Army Campaign Plan for the Future Force vision and the expected approaches to develop an acquisition strategy to achieve that vision. The results from the present workshop, stressing the role of requirements management in this larger system-of-systems context, are directly related to results from the Future Force Workshop.

6 Summary

This report summarizes the results and analyses of a workshop dealing with requirements issues in a system-of-systems context. Senior staff associated with the Army PEO Aviation and TRADOC Directorate of Combat Developers attended the workshop.

One theme that clearly emerges from this work is the following: *There is a conflict in the approach to a system of systems between top-down, policy driven and bottom-up, program-centric approaches.* This clash is perhaps expected: the acquisition community is naturally focused on a particular system, and there are mandates about how funding is allocated in this manner. An SoS requires a larger perspective, however. Based on this workshop, the clash was evident from the perspective of requirements management in a system-of-systems context.

From an analysis of issues presented at this workshop, there are a number of possible follow-on efforts. Among them, we would mention:

- exploring the role of the Army software blocking policy, as it relates to achieving interoperability among requirements
- investigating the role of the JCIDS process and its relation to requirements management implemented by a particular service-specific system(s) in the context of an SoS
- investigating the role of the PPBES as it relates to the acquisition of an SoS, as compared to the acquisition of individual systems
- investigating the possible barriers imposed by the current legal framework (laws, regulations, policies, etc.) that impact acquisition of an SoS
- understanding the differences between an SoS and an FoS, as per the currently specified Army acquisition process (recall the discussion on Section 1)
- investigating the approach of using patterns (and antipatterns) as a means to make and present inferences regarding user-collected information from workshops such as this one

There is work ongoing in some of these topics at the SEI. There is also a need to integrate the results of this workshop with related efforts to gain a broader understanding of interoperability aspects of system-of-systems management, construction, and operation. This workshop is a step along the way to developing, and understanding, the larger picture.

A Family of Systems and System of Systems

In the introduction, a definition of the term *system of systems* was presented. A related term is a *family of systems*. Our purpose in this Appendix is to examine the manner in which these terms are defined. As background, both terms appeared in connection with the *Requirements Generation System*. In that document, those terms are defined as follows:

> system of systems: A set or arrangement of systems that are related or connected to provide a given capability. The loss of any part of the system will degrade the performance or capabilities of the whole.
>
> family of systems: A set or arrangement of independent systems that can be arranged or interconnected in various ways to provide different capabilities. The mix of systems can be tailored to provide desired capabilities dependent on the situation [JCS 01]

The Army adopted these definitions with some further explanatory text. However, it is interesting to note that the *Army Software Blocking Policy* (published in 2001) contains the following:

> system of systems: The collection of systems that share/exchange information which interact synergistically. Other documents outside of this policy may refer to the SoS as a 'Family of Systems' [DoA 01]

Later, the *Army Acquisition Policy*,[10] published in 2003, includes the following definitions:

> system of systems: A set or arrangement of interdependent systems that are related or connected to provide a given capability. The loss of any part of the system will degrade the performance or capabilities of the whole. An example of a SoS could be interdependent information systems. While individual systems within the SoS may be developed to satisfy the peculiar needs of a given user

[10] The Army Acquisition Policy is principally focused on a system of systems and only mentions a family of systems. The references to a system of systems and a family of systems number 21 and 4, respectively. Apart from the definition of a family of systems, the only substantive comment is that a Capstone Requirements Document applies to a family of systems.

group, the information they share is so important that the loss of a single system may deprive other systems of the data needed to achieve even minimal capabilities.

family of systems: A set or arrangement of interdependent systems that can be arranged or interconnected in various ways to provide different capabilities. The mix of systems can be tailored to provide desired capabilities, dependent on the situation. An example of a family of systems is a unit of action that included armor, infantry, artillery, and combat support systems [DoA 03].

Further explanation regarding a family of systems may be found in the *Operation of the Joint Capabilities and Integration Development System*, published in 2004, where the following example is added to the definition:

An example of a FoS would be an anti-submarine warfare FoS consisting of submarines, surface ships, aircraft, static and mobile sensor systems and additional systems. Although these systems can independently provide militarily useful capabilities, in collaboration they can more fully satisfy a more complex and challenging capability: to detect, localize, track and engage submarines [JCS 04].

Still more discussion of a both terms may be found in the *Joint Capabilities and Integration Development System*, published in 2005:

system of systems (SoS): A set or arrangement of interdependent systems that are related or connected to provide a given capability. The loss of any part of the system will significantly degrade the performance or capabilities of the whole. The development of a[n] SoS solution will involve trade space between the systems as well as within an individual system performance. An example of a[n] SoS would be a combat aircraft. While the aircraft may be developed as a single system, it could incorporate subsystems developed for other aircraft. For example, the radar from an existing aircraft may be incorporated into the one being developed rather than developing a new radar. The SoS in this case would be the airframe, engines, radar, avionics, etc. that make up the entire combat aircraft capability.

family of systems (FoS): A set of systems that provide similar capabilities through different approaches to achieve similar or complimentary effects. For instance, the warfighter may need the capability to track moving targets. The FoS that provides this capability could include unmanned or manned aerial vehicles with appropriate sensors, a space-based sensor platform or a special operations capability. Each can provide the ability to track moving targets, but with differing characteristics of persistence, accuracy, timeliness, etc [JCS 05].

It is not our intent to engage in a detailed discussion of the differences between these concepts. As noted on page 1, the attendees did not distinguish between an SoS and an FoS during the workshop.

However, one perspective on the difference between an SoS and an FoS can be addressed in terms of the coupling by the individual components that form the larger unit. We would suggest, for example, there is a tighter coupling among components of an SoS than an FoS. One could also argue that an FoS is not a system, but rather a collection of systems built to a common set of rules and standards that can be composed to form an SoS.

A point here is that to the extent that there may be differences between an SoS and an FoS, there can be implications for the requirements process. This point is discussed further in Appendix B.

B Requirements for Systems of Systems: An Alternative View

The definition provided on page 1 essentially labels an SoS as a large-scale entity wherein each of its components is a system. Presumably the system and the SoS can be developed using traditional system engineering techniques. However, such a definition leads the reader into thinking that an SoS is really just a larger scale object than a system. An alternative view provided by Maier[11], is that an SoS is of a different nature. Maier states that an SoS has the following characteristics:

- *operational independence of the systems*
  Each of the individual systems within an SoS has a "life of its own" and can function acceptably and provide useful service without necessarily interacting with other systems.
- *managerial independence of the systems*
  The individual systems within an SoS are under different authorities. Within the DoD context, these could be significantly different authorities in that different services will own different systems in the context of an SoS.
- *evolutionary development*
  The different systems within the SoS are developed and upgraded on uncoordinated schedules. While policies such as the software blocking policy can coordinate the schedules for a number of systems within an SoS, it is unlikely that such a policy can scale to a size of the Global Information Grid (GIG).
- *emergent behavior*
  The behavior of the SoS as a whole is not embodied in any one of the systems within it. Emergent behavior is a direct consequence of having the systems interact; the difficulty is ensuring that the emergent behavior is desirable.
- *geographic distribution*
  Simply put, the systems within the SoS are not all co-located. While it is highly likely that any significant fielded SoS *will* have this characteristic, it is by no means obvious that it is a necessary characteristic.

[11] A reference to Maier's work, as discussed here, can be found at http://www.infoed.com/Open/PAPERS/systems.htm.

In a more recent presentation, Maier no longer discusses the emergent behavior characteristic and suggests that there are counter examples to the geographic distribution aspect. Hence, geographic distribution is no longer considered a fundamental characteristic.

Notice that the definition of an SoS, provided in the Introduction and elaborated further in Appendix A, is not described in terms of the characteristics identified by Maier. One might infer, for example, that the approach to an SoS in terms of *interdependent* systems would disagree with the Maier characteristic of operational independence.

However, our intent here is not to examine the current definitions in light of the Maier characteristics. We intend to consider, based on the three key characteristics (namely, operational independence, managerial independence, and evolutionary development), what it means to perform requirements engineering in the context of an SoS.

It is clear that each of the individual systems will have a definition of the requirements for its own independent operational capabilities. However, neither the origin of requirements for the *emergent behavior* nor the way those requirements are levied on individual systems is obvious. Emergent behavior, as we stated, is not a result of the operation of any one system; rather, it is the result of the interactions between multiple systems. No single system can be tested to see if it exhibits the emergent behavior. It is only in the context of an SoS that a test can be performed. While they can demonstrate the presence of desirable emergent behavior, such tests are unlikely to demonstrate the absence of undesirable emergent behavior. In the extreme case, where systems are composed dynamically, no *a priori* test is possible when the SoS is assembled—since, at that time, no one can know which systems will be called upon to satisfy any given mission thread.

Within an SoS, each individual system will likely have a different owner, and those owners will report up through independent management chains! Potentially, in a large SoS, those chains may report in different services, departments, or even coalition partners. It would be nice if some group were responsible for the development of requirements of the SoS, including its emergent behavior. Yet, since no one group owns the SoS, there is no obvious group that owns its requirements.[12] And, again, in the extreme case, those requirements may only be known when a particular mission requires functionality from a particular set of systems within the SoS.

Finally, if we consider the issue of evolutionary development, we again see a problem for requirements. Since the various systems within the SoS will change at

their own rates (the software blocking policy notwithstanding), the capabilities of the individual systems will also change independently. Indeed, in the more likely case, the systems that compose the SoS will change over time as systems come and go. Thus, the capabilities of the SoS can only ever be defined in terms of what it can do at any given instant, and those capabilities are expected to evolve over time. If the capabilities are unknowable at assembly time, it is also unknown, then, which requirements can or cannot be satisfied.

Now, the above arguments are not an attempt to dismiss requirements for an SoS as being impossible to develop or maintain. Rather, they are the basis for suggesting that a new approach to system-of-system requirements will be necessary. Traditional system engineering techniques do not appear to apply to the SoS; they are necessary, but not sufficient. Unfortunately, it is not immediately obvious how the problem can or should be solved.

[12] In addition, it would be nice if there were a central group responsible for the identification and resolution of conflicts among requirements that may exist. However, other approaches to conflict identification and resolution are also possible. For example, it may be the responsibility of the individual systems to identify and resolve conflicts based upon peer interactions.

C Requirements Management

Requirements management is a key process that is performed as part of the acquisition of a system. The process is discussed in a number of sources, such as the SEI CMMI®[13] framework. A typical list of activities associated with requirements management includes the following (from CMMI) [Chrissis 03]:

- develop customer requirements
  - collect stakeholder needs
  - elicit needs
  - develop the customer requirements
- develop product requirements
  - establish product and product-component requirements
  - allocate product-component requirements
  - identify interface requirements.
- analyze and validate requirements
  - establish operational concepts and scenarios
  - establish a definition of required functionality
  - analyze requirements
  - analyze requirements to achieve balance
  - validate requirements
  - validate requirements with comprehensive methods
- manage requirements
  - obtain an understanding of requirements
  - obtain commitment to requirements
  - manage requirements changes
  - maintain bidirectional traceability of requirements
  - identify inconsistencies between project work and requirements

[13] CMMI is registered in the U.S. Patent and Trademark Office by Carnegie Mellon University.

It is important to note that the list above is implicitly specified for the context of a particular system. It is not necessarily intended to be applicable to an SoS, although such a perspective may be (and often is) taken.

D Software Blocking

None of the issues identified by the attendees explicitly referred to the Army software blocking policy. However, there was discussion of the Army software blocking during the workshop. It is clear that the blocking policy plays an important role in facilitating the development of an Army approach to an SoS. As one attendee said, “Software blocking has become the agent for interoperability.” The SWB policy must be implemented by all PEOs and PMs. Some background on software blocking is described in the *Army Acquisition Policy* and the *Army Software Blocking Policy* documents [DoA 03 and DoA 01].

The purpose of the software blocking process is to facilitate the development and sustainment of systems-of-systems interoperability through the periodic delivery of a collection of elements to the operational community. It can be viewed as an approach to the integration of multiple systems. The term *block* is used to describe a process having two segments:

1. *preparation*
   During this segment products related to requirements and architecture are developed. A key product is the development and approval of the Block Execution Management Plan (BEMP). The segment is expected to take 18 months to complete.
2. *software development*
   The software development segment begins after the BEMP has been approved and is managed according to the BEMP. The segment is complete when the software development for the systems has been completed and certification of the block (including interoperability and system-of-systems evaluation) has been passed. Nominally, this segment is expected to take 36 months to complete.

A *software block* refers to a software baseline which is designed to satisfy the set of system-of-system requirements applicable to the block. Software blocks may overlap to provide software updates fielded every 18 months. The nature of requirements is expected to be addressed in the preparation phase of the block.

There is a relation between an SoS and a software block.[14] For example, as we noted on page 20, "system engineering must be done from a system-of-systems perspective in accordance with the software blocking policy and its implemented processes."

During discussion of the blocking process, several points were noted by the attendees, including:

- Software blocking is a voluntary (unfunded) process and there is a lack of enforcement and accountability.
- Software blocking has not focused on requirements development.
- There is no single point of authority for control of a block, couched as the lack of a "trail boss."
- There is no Milestone Decision Authority (MDA) for a block.
- Who is in charge of writing requirements for an SoS?
- It is not clear that there is a system engineer for a block. The closest organization for this would be G6, but they might not have the knowledge and time to perform the activity.
- Software blocking operates in a pair-wise manner between systems to negotiate interoperability. This approach is better for hardware than with data and operational semantics that apply to the software.[15]

No doubt, some of the points raised may reflect the relative newness of the application of the software blocking policy. However, all of these issues are consistent with ones identified earlier in the workshop.

It is not the intent of this report to go into details of the Army software blocking policy per se or the manner in which it is implemented. However, some of the issues identified have a direct bearing on the concepts associated with a software block. In particular, we refer to the issues dealing with a lack of system engineering and requirements management at a level above a program and the lack of system-of-systems policies and funding streams. A software block must address

---

[14] We make no distinction here between a system of systems and a family of systems, which was brought up in Section 1 and discussed further in Appendix A.

[15] An example of the difficulties that may arise was pointed out by an attendee concerning the development of a *Combat Status Report*. Consider the amount of fuel left in an aircraft. One participant would prefer that this information be displayed by a simple, color-coded scheme of red, yellow, and green. Another participant would prefer to see a quantitative measure, such as how many gallons of fuel are left or how long the fuel will last. These preferences represent different interpretations of the same data by different user communities.

issues just such as these. Further study of the software blocking policy is ongoing by SEI staff with special emphasis on lessons learned from the execution of the first increment of a software block.

E Discussion of Pattern Development

Much of the post-workshop analysis was based on development of patterns and antipatterns, terms discussed in Section 4. This Appendix provides further discussion and an example of a pattern development.

E.1 Discussion

Because we sought a simple, systematic form to represent issues, we chose to use patterns and antipatterns. Representing a pattern in the form $<A_1, A_2; R>$, where A_1 and A_2 represent actors, and R describes the relation between them,[16] provided the necessary notation.

Recall that the issues were elicited in the form of a condition-consequence pair. Both the condition and consequence aspects of an issue can be represented as patterns. Hence, to the extent that a condition *implies* a consequence, that implication may be carried over to the representation of a pattern implying another pattern.

The process of developing patterns often involved dealing with compound statements. As an example, consider the statement "System_X and System_Y interoperate with System_Z." It is obvious that this statement is really the combination of two more basic patterns, namely <System_X, System_Z; interoperate> and <System_Y, System_Z; interoperate>.

Of special importance to the workshop was the fact that we sought to identify issues raised by the attendees. The use of antipatterns is especially relevant when textual statements are couched as issues. That is, an issue typically includes a statement that is critical of something or indicates the absence of something. For example, the issue "The PMO does not perform system engineering" asserts a negation of the PMO performing system engineering. Glancing over the list of

[16] The relation among actors is typically expressed as a verb. Sometimes in presenting the patterns, or their use in a context map, a shorthand notation can be used. For example, the statement: "The system requirements are not clear" would be represented in a shorthand form: <system, requirements, clear>. Thus, for simplicity we do not include the more correct are clear as it is implied.

issues in Section 3, you will recognize the importance of the use of antipatterns to the discussion of issues collected at the workshop. In fact, it is issues, represented as antipatterns, that often provide very useful information.

We emphasize again our need to have a simple, systematic form for the representation of issues identified by the attendees. The use of patterns proved to meet that need.[17] The fact that there is a direct relation between a pattern and some aspect of a context map proved all the more valuable in the development of themes involving multiple issues.

E.2 Example

The following illustrates the development of patterns based on issues identified by the attendees. We will only consider the case of those issues related to joint considerations (discussed in Section 4.5). The development of other patterns is similar.

Issue 10 identified at the workshop was stated: "The JCIDS process has no path that leads to a view (architecture) that can be used for a statement of specification for a material developer or test criteria by the combat developer; there is no direct link from requirements to end product." From this text, the following patterns were developed:

<architecture, JCIDS process; results from>
<Material Developer, architecture; uses>
<Material Developer, specification; creates>
<Combat Developer, architecture; uses>
<Combat Developer, test criteria; creates>
<test criteria, testing; used in>

Notice that in developing these patterns it was necessary to also include a pattern to indicate that test criteria are used in testing. This pattern will be used to help integrate the pattern from issue 22, discussed below. Recall that the presence of an underscore denotes negation of the indicated expression.

Issue 18 was "There is a lack of Joint Vision (e.g., NCOW-RM) and a system-of-systems organization structure; leads toward a stovepipe development."

[17] There is more that can be done for a pattern approach. Transforming a textual statement into the logical conjunction of the (more basic) patterns is but one example. We have not attempted to bring a more formal approach (e.g., grammar and calculus) to the use of patterns constructed in this workshop.

The following patterns are identified:

<Joint Vision, SoS; applies to>
<SoS, SoS Organization structure, based on>
<Stovepipe, Organizational structure; type of>

Issue 22 was "Joint system-of-systems requirements are not clear; interoperability is not guaranteed and joint testing results are questionable." The following patterns were developed:

<SoS, Joint Requirements; clear>
<SoS, interoperability; guaranteed>
<SoS, joint test results; questionable>

When the preceding patterns are combined, we have the overall diagram shown in Figure 14. The dark boxes with white text are related to condition clauses associated with the issues. Normally, a context diagram will not be presented in this highlighted manner; it is done here simply as an aid to the reader. Further discussion of this diagram appears in Section 4.5 beginning on page 20.

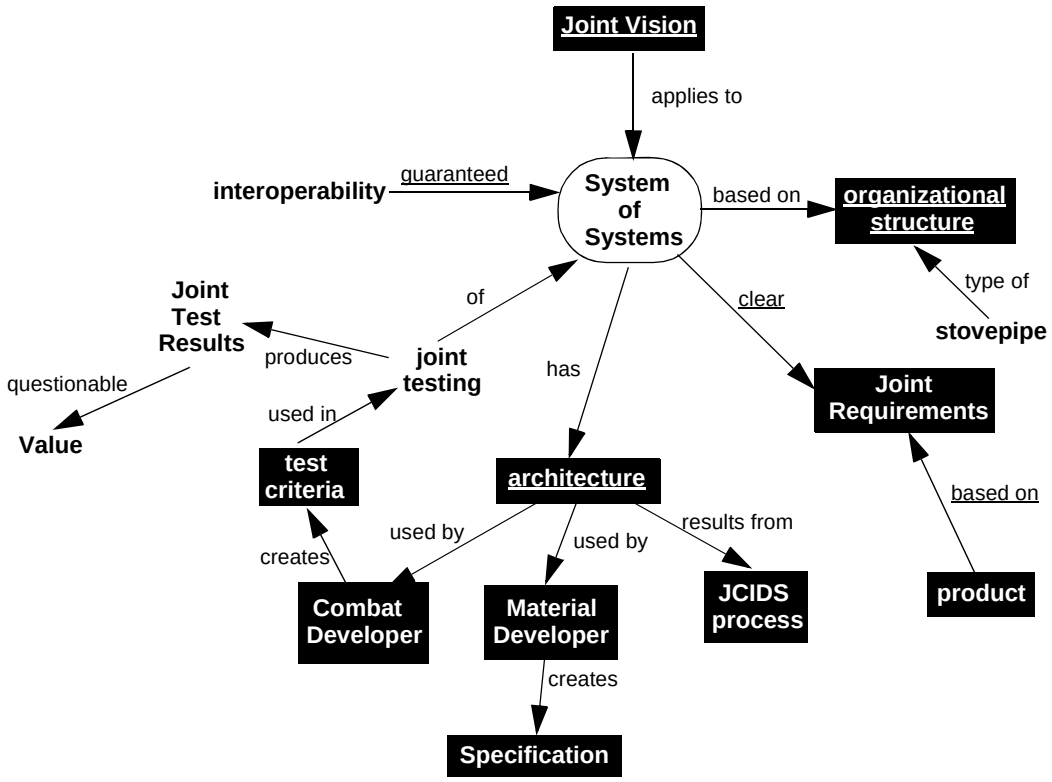


Figure 14: Summary of Joint Issue Relations

F Acronyms

A2C2S	Army Airborne Command and Control System
AOA	Analysis of Alternatives
ASA/ALT	Assistant Secretary of the Army for Acquisition, Logistics, and Technology
BEMP	Block Execution Management Plan
CMMI	Capability Maturity Model Integration
COTS	Commercial off-the-shelf
DCD	Directorate of Combat Developments
FAR	Federal Acquisition Regulations
FoS	Family of Systems
G3	Army Deputy Chief of Staff, responsible for development of policy and guidance for requirements, including operational requirements process.
G5	Army Recruiting Command Directorate
G6	Office of the Army Chief Information Officer
G7	Army Materiel Command
G8	Army Deputy Chief of Staff, responsible for programming, materiel integration, studies and analyses, and external reviews.
ICD	Initial Capabilities Document
ISIS	Integration of Software-Intensive Systems
JCIDS	Joint Capabilities Integration and Development System
JROC	Joint Requirements Oversight Council
MDA	Milestone Decision Authority
NCOW-RM	Network Centic Operations and Warfare — Reference Model
OSD	Office of the Secretary of Defense
PEO	Program Executive Office
PPBES	Planning, Programming, Budgeting, and Execution System
PM	Program Manager
PMO	Program Management Office
TOC	Tactical Operations Center

PMO	Program Management Office
SEMP	System Engineering Master Plan
SoS	System of Systems
SoSAMO	System-of-Systems Acquisition Management Organization
SOSI	System-of-Systems Interoperability
SoSRMO	System of Systems Requirements Management Organization
SWB	Software Blocking
TOC	Tactical Operations Center
TPIO	TRADOC Program Integration Office
TRADOC	Training and Doctrine Command
TSM	TRADOC System Manager
USAAWC	U. S. Army Aviation Warfighting Center

References

URLs are valid as of the publication date of this document.

Chrissis 03] Chrissis, Mary Beth; Konrad, Mike; & Shrum, Sandy. *CMMI: Guidelines for Process Integration and Product Improvement.* Boston, MA: Addison Wesley, 2003.

[DoA 01] U.S. Department of the Army. *Army Software Blocking Policy,* Version 11.4E. U. S. Department of the Army, Washington, D.C.: September 18, 2001. jitc.fhu.disa.mil/vmf_conf/2004_jul/docs /010918_armyblocking_approved.doc

[DoA 03] U.S. Department of the Army. *Army Acquisition Policy,* Army Regulation 70-1. U.S. Department of the Army, Washington, D.C.: December 31, 2003. http://www.army.mil/usapa/epubs/pdf /r70_1.pdf

[JCS 01] Chairman of the Joint Chiefs of Staff. *Requirements Generation System,* CJCSI 3170.01B. U.S. Department of Defense, Washington, D.C.: April 15, 2001. http://www.dtic.mil/doctrine/jel /cjcsd/cjcsi/3170_01b.pdf

[JCS 03] Chairman of the Joint Chiefs of Staff. *Joint Capabilities Integration and Development Process,* CJCSI 3170.01C. U.S. Department of Defense, Washington, D.C.: June 24, 2003. http:// www.army.mil/howwewillfight/references/8%203170_01c.pdf

[JCS 04] Chairman of the Joint Chiefs of Staff. *Operation of the Joint Capabilities Integration and Development System,* CJCSI 3170.01A. U.S. Department of Defense, Washington, D.C.: March 12, 2004. http://www.aiaa.org/tc/sos/ Launch_Management_Guide/Appendix%20B%5CICD_CDD_CPD.pdf

[JCS 05] Chairman of the Joint Chiefs of Staff. *Joint Capabilities Integration and Development System,* CJCSI 3170.01E. U.S. Department of Defense, Washington, D.C.: May 11, 2005. http://www.dtic.mil/cjcs_directives/cdata/unlimit/3170_01.pdf

[Levine 03] Levine, Linda; Meyers, B. Craig; Morris, Ed; Place, Patrick R. H.; & Plakosh, Daniel. *Proceedings of the System of Systems Interoperability Workshop (February 2003)* (CMU/SEI-2003-TN-016, ADA416429). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 2003. http://www.sei.cmu.edu/publications/documents/03.reports/03tn016.html

[Morris 04] Morris, Ed; Levine, Linda; Meyers, B. Craig; Place, Patrick R. H.; & Plakosh, Daniel. *System of Systems Interoperability (SOSI); Final Report.* (CMU/SEI-2004-TR-004). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 2004. http://www.sei.cmu.edu/publications/documents/04.reports/04tr004.html

[Smith 05] Smith, James D. & Meyers, B. Craig. *Exploring Programmatic Interoperability: Army Future Force Workshop* (CMU/SEI-2005-TN-042). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 2005. http://www.sei.cmu.edu/publications/documents/05.reports/05tn042.html

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY (leave blank)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	March 2006	Final

4. TITLE AND SUBTITLE	5. FUNDING NUMBERS
Requirements Management in a System of Systems Context: A Workshop	FA8721-05-C-0003

6. AUTHOR(S)
B. Craig Meyers, James D. Smith, Peter Capell, and Patrick R. H. Place

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)	8. PERFORMING ORGANIZATION REPORT NUMBER
Software Engineering Institute Carnegie Mellon University Pittsburgh, PA 15213	CMU/SEI-2006-TN-015

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSORING/MONITORING AGENCY REPORT NUMBER
HQ ESC/XPK 5 Eglin Street Hanscom AFB, MA 01731-2116	

11. SUPPLEMENTARY NOTES

12.a DISTRIBUTION/AVAILABILITY STATEMENT	12.b DISTRIBUTION CODE
Unclassified/Unlimited, DTIC, NTIS	

13. ABSTRACT (maximum 200 words)

This report summarizes the results of a workshop focused on requirements management in a system of systems. The workshop attendees were affiliated with the Army Program Executive Office (PEO) Aviation and Training and Doctrine Command (TRADOC) Combat Developers. During the workshop, issues were identified in a number of areas, including requirements management, system-of-systems management, and system construction. Many of the issues raised address some form of the conflict that exists between a top-down, policy driven approach to the acquisition of a system of systems and a bottom-up, program-centric approach to the acquisition of an individual system.

14. SUBJECT TERMS	15. NUMBER OF PAGES
system of systems; requirements management; interoperability; programmatic interoperability	67
	16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	UL